

Contents

1 Routine/Function Prologues	2
1.1 Fortran: Module Interface time_interp_agrmet.F90 (Source File: time_interp_agrmet.F90)	2

1 Routine/Function Prologues

1.1 Fortran: Module Interface time_interp_agrmet.F90 (Source File: time_interp_agrmet.F90)

This routine consists of utilities for temporal interpolation of radiation forcing
 TIME1 = most recent past data TIME2 = most recent future data

REVISION HISTORY:

```

28 Oct 1999: Brian Cosgrove; Initial code, see getrad.f
27 Apr 2000: Brian Cosgrove' Disabled zenith angle correction cutoff for
             cos(zen) less than .2
11 May 2000: Brian Cosgrove; Enabled correction cutoffs for cos(zen) less
             than .1, stop model if computed value is greater than 1367 w/m2
08 Jan 2001: Brian Cosgrove; Added check to see if czb or czm is equal
             to zero before trying to divide by czm or czb. If it is
             zero, set radiation value to zero
06 Mar 2001: Brian Cosgrove; Changed computation of WT1 and WT2 in cases
             where the previous hour or the next hour of observed radiation
             is not available. Substituted TIME1 for LDAS%PINKTIME1 and
             LDAS%NESTIME1 and TIME2 for LDAS%PINKTIME2 and LDAS%NESTIME2
15 Jun 2001: Urszula Jambor; Reworked algorithm for AGRMET data & GLDAS.
15 Oct 2001: Jesse Meng; Replace lis%agrmet flag by lis%agrmetsw and
             lis%agrmetlw; added call retagrlw() to calculate
             AGRMET LW;
25 Feb 2002: Urszula Jambor; check on both SW & LW file status before
             using
04 Jun 2002: Urszula Jambor; allowed fall back to model SW.
10 Dec 2002: Urszula Jambor; replaced lis%astat1,2 with local sstat1,2
             Reorganized routine to mirror other get-routines, and
             corrected bug in file status check for initial time1.

```

INTERFACE:

```
subroutine time_interp_agrmet
```

USES:

```

use lisdrv_module, only : lis, grid
use obsradforcing_module, only : obswdata1, obswdata2, oblodata1, oblodata2, &
      sstat1, sstat2, lstat1, lstat2
use grid_spmdMod
use time_module
use agrmetdomain_module, only : agrmetdrv
implicit none

```

CONTENTS:

```

!-----
! Loop through and replace data as possible with AGRMET data
! This depends on options specified in lis.crd as well as actual

```

```

! data holdings.
!-----
allocate(obsw(gdi(iam)))
yr1 = lis%t%yr      !Previous Hour
mo1 = lis%t%mo
da1 = lis%t%da
hr1 = lis%t%hr
mn1 = 0
ss1 = 0
ts1 = 0
call tick ( time1, doy1, gmt1, yr1, mo1, da1, hr1, mn1, ss1, ts1 )

yr2 = lis%t%yr      !Next Hour
mo2 = lis%t%mo
da2 = lis%t%da
hr2 = lis%t%hr
mn2 = 0
ss2 = 0
ts2 = 60*60
call tick ( time2, doy2, gmt2, yr2, mo2, da2, hr2, mn2, ss2, ts2 )
#if(defines SPMD)
call MPI_BCAST(agrmtdrv%agrmttime1,1,MPI_REAL8,0,&
    MPI_COMM_WORLD,ierr)
call MPI_BCAST(agrmtdrv%agrmttime2,1,MPI_REAL8,0,&
    MPI_COMM_WORLD,ierr)
call MPI_BCAST(lis%t%time,1,MPI_REAL8,0,&
    MPI_COMM_WORLD,ierr)
call MPI_BCAST(lis%t%gmt,1,MPI_REAL,0,&
    MPI_COMM_WORLD,ierr)
#endif
if ((sstat1 == 1) .and. (sstat2 == 1)) then
!-----
! Compute weights and zenith angle information. Replace forcing
! with AGRMET data.
!-----
wt1 = (agrmtdrv%agrmttime2 - lis%t%time) / (agrmtdrv%agrmttime2 - agrmtdrv%agrmttime1)
wt2 = 1.0 - wt1
do c=1,gdi(iam)
    zdoy = lis%t%doy
    call zterp ( 1, grid(c)%lat, grid(c)%lon, gmt1, gmt2, &
        lis%t%gmt, zdoy, zw1, zw2, czb, cze, czm, lis)
    obsw(c) = lis%d%udef
    if ((obswdata1(c)>0.0) .and. (obswdata2(c)>0.0)) then
        obsw(c) = obswdata1(c)*zw1+obswdata2(c)*zw2
        if ((obsw(c) > obswdata1(c) .and. &
            obsw(c) > obswdata2(c)) .and. &
            (czb<0.1 .or. cze<0.1)) then
            obsw(c) = obswdata1(c)*wt1+obswdata2(c)*wt2

```

```

        end if
    end if
    if ((obswdata1(c) > 0.0) .and. &
        (obswdata2(c) <= 0.0)      ) then
        if (czb > 0.0) then
            obsw(c) = obswdata1(c) * (czm/czb)
        else
            obsw(c) = obswdata1(c) * (0.0)
        end if
        if ((obsw(c) > obswdata1(c) .and. &
            obsw(c) > 0.0) .and. (czb<0.1 .or. cze<0.1)) then
            obsw(c) = obswdata1(c)*wt1 + 0.0*wt2
        end if
    end if
    if ((obswdata1(c)<=0.0) .and. (obswdata2(c)>0.0)) then
        if (cze > 0.0) then
            obsw(c) = obswdata2(c) * (czm/cze)
        else
            obsw(c) = obswdata2(c) * (0.0)
        end if
        if ((obsw(c)>0.0 .and. obsw(c)> obswdata2(c)) &
            .and. (czb < 0.1 .or. cze < 0.1)) then
            obsw(c) = 0.0*wt1 + obswdata2(c)*wt2
        end if
    end if

    if (obsw(c) >= 0.0) then
        if (obsw(c) .gt. 1367) then
            print *, 'FALLING BACK TO MODEL SW'
        else
            grid(c)%forcing(3) = obsw(c)
        end if
    end if
    end do !c
end if

if ((sstat1 == 1) .and. (sstat2 /= 1)) then
!-----
! Compute weights and zenith angle information. Replace forcing
! with zenith extrapolated AGRMET data
!-----
    wt1 = (time2 - lis%t%time) / (time2 - agrmetdrv%agrmtim1)
    wt2 = 1.0 - wt1
    do c=1, gdi(iam)
        zdoy = lis%t%doy
        call zterp ( 1, grid(c)%lat, grid(c)%lon, gmt1, gmt2, &
                    lis%t%gmt, zdoy, zw1, zw2, czb, cze, czm, lis)
        obsw(c) = lis%d%udef

```

```

      if (obswdata1(c) > 0.0) then
        if (czb > 0.0) then
          obsw(c) = obswdata1(c) * (czm/czb)
        else
          obsw(c) = obswdata1(c) * (0.0)
        end if
        if ((obsw(c) > 400.0) .and. &
            (czb < 0.1 .or. cze < 0.1) ) then
          obsw(c) = obswdata1(c)*wt1
        end if

        if (obsw(c) >= 0.0) then
          if (obsw(c) .gt. 1367) then
            print *, 'warning, OBSERVED SW RADIATION HIGH'
            print *, 'it is',grid(c)%forcing(3), ' at ',c
            print *, 'agrm1=',obswdata1(c)
            print *, 'agrm2=',obswdata2(c)
            print *, 'wt1,wt2,czb,cze,czm,zw1,zw2'
            print *, wt1,wt2,czb,cze,czm,zw1,zw2
            print *, 'FALLING BACK TO MODEL SW'
          else
            grid(c)%forcing(3) = obsw(c)
          end if
        end if
      end if
    end do !c
  end if

  if ((sstat1 /= 1) .and. (sstat2 == 1)) then
!-----
! Compute weights and zenith angle information. Replace forcing
! with zenith extrapolated AGRMET data
!-----
    wt1 = (agrmetdrv%agrmtime2 - lis%t%time) / (agrmetdrv%agrmtime2 - time1)
    wt2 = 1.0 - wt1
    do c=1, gdi(iam)
      zdoy = lis%t%doy
      call zterp ( 1, grid(c)%lat, grid(c)%lon, gmt1, gmt2, &
                  lis%t%gmt, zdoy, zw1, zw2, czb, cze, czm, lis)
      obsw(c) = lis%d%udef
      if (obswdata2(c) > 0.0) then
        if (cze > 0.0) then
          obsw(c) = obswdata2(c) * (czm/cze)
        else
          obsw(c) = obswdata2(c) * (0.0)
        end if
        if ((obsw(c) > 400.0) .and. &
            (czb < 0.1 .or. cze < 0.1) ) then

```

```
      obsw(c) = 0.0*wt1 + obswdata2(c)*wt2
      end if

      if (obsw(c) >= 0.0) then
          if (obsw(c) .gt. 1367) then
          else
              grid(c)%forcing(3) = obsw(c)
          end if
      end if
      end if
      end do !c
  end if
  if ((sstat1 /= 1) .and. (sstat2 /= 1)) then
      do c=1, gdi(iam)
          obsw(c) = lis%d%undef
      end do
  end if
  if ((lstat1 == 1) .and. (lstat2 == 1)) then
      wt1 = (agrmetdrv%agrmtime2 - lis%t%time) / &
      (agrmetdrv%agrmtime2 - agrmetdrv%agrmtime1)
      wt2 = 1.0 - wt1
      do c=1,gdi(iam)
          if ( (oblwdata1(c) > 0.0) .AND. &
              (oblwdata2(c) > 0.0)) then
              grid(c)%forcing(4)= oblwdatal(c)*wt1 &
                  + oblwdatal(c)*wt2
          end if
      end do
  end if
  deallocate(obsw)
  return
```